Looking at a class of RFID APs through GNY logic

Xiaowen Zhang*
Department of Computer Science,
College of Staten Island,
City University of New York,
2800 Victory Blvd, Staten Island, NY 10314, USA
Fax: 1-718-982-2856
E-mail: xiaowen.zhang@csi.cuny.edu
*Corresponding author

Qinghai Gao
Department of Security Systems,
Farmingdale State College/SUNY,
2350 Broadhollow Road, Farmingdale, NY 11735, USA
E-mail: gaoqj@farmingdale.edu

Mohamed K. Saad
Graduate Center,
Department of Computer Science,
City University of New York,
365 Fifth Ave, New York, NY 10016, USA
E-mail: msaad@gc.cuny.edu

Abstract: Radio Frequency Identification Authentication Protocols (RFID APs) are an active research topic and many protocols have been proposed. In this paper, we consider a class of recently proposed lightweight RFID authentication protocols: CRAP, LCAP, OHLCAP, O-TRAP, YA-TRAP, and YA-TRAP+, which are claimed to be resistant to conventional attacks and suitable for low cost RFID device scenarios. We examine them using GNY logic to determine whether they can be proved to have achieved their protocol goals. We show that most of them meet their goals, though some do not. Furthermore this approach enables us to identify similarities and subtle differences among these protocols. Finally, we offer guidelines on when it is necessary to use encryption rather than hash functions in the design of RFID authentication protocols.

Keywords: RFID; authentication protocol; GNY logic; formal method; CRAP; LCAP; OHLCAP; O-TRAP; YA-TRAP; hash; encryption; network security.


Biographical notes: Xiaowen Zhang is an Assistant Professor of Computer Science at College of Staten Island (CSI). Prior to joining CSI, he worked both in Academia as a Research Fellow and Lecturer, as well as in industry as a Software and Electronic Engineer. His research interests include information security, cryptography, RFID, quantum computing, and wireless communications. He received a PhD in Computer Science from the City University of New York (CUNY), New York, USA in 2007, and a PhD in Electrical Engineering from Northern Jiaotong University, Beijing, China in 1999.

Qinghai Gao is a Faculty Member of the Department of Security Systems at Farmingdale State College, SUNY. Before joining Farmingdale, he worked as an Information Security Specialist in industry for a few years. His present research interests include digital forensics, network security, identity theft, biometrics, biological information system, DNA computing, cryptography and steganography.
1 Introduction

An RFID system usually consists of three kinds of components: many RFID tags (or transponders), several RFID readers (or interrogators), and a few backend servers. An RFID tag is a tiny microchip equipped with a radio frequency antenna. It is capable of exchanging the tag identification and other related data with the reader through radio frequency wireless media. A reader is an electronic device located between the tags and a backend server. A reader gets data from or sends data to the tag; meanwhile the reader communicates with a backend server. The backend server runs application software, hosts databases, and processes tag data received from a reader. A server communicates (through wireless or wired connections) with readers on one end and with the enterprise network (e.g., the internet) infrastructure on the other end. The wireless communication links between tags and readers are considered vulnerable to security and privacy threats.

Security and privacy threats are, but not limited to, eavesdropping, forging, tampering, shielding, jamming, and spoofing. Adding security features to RFID systems, especially low-cost tags, is a challenging task because they are extremely resource limited and cannot support computationally expensive cryptographic algorithms. Current research on possible RFID security algorithms include lightweight public-key cryptography (Anshel et al., 2006; Vaudenay, 2006; McLoone and Robshaw, 2007), provably secure lightweight hash functions like SQUASH (Shamir, 2008) among others, and provably secure authentication protocols such as the HB family (Hopper and Blum, 2001; Juels and Weis, 2005; Hammouri and Sunar, 2008). Practical RFID authentication protocols should be lightweight, ensure anonymity (un-traceability), defeat impersonation, denial of service, man-in-the-middle attacks. As documented in the literature (Avoine, 2005; Juels et al., 2008; Sarma et al., 2002), RFID and security experts have devoted a great deal of effort to satisfy these requirements. Among those efforts, new RFID authentication protocols and analysis are active areas of research (Chatmon et al., 2006; Gilbert et al., 2005; Juels, 2004; Juels and Weis, 2005; Le et al., 2007; Li and Deng, 2007; Peris-Lopez et al., 2006). As a result, tens of RFID authentication protocols have been published in scientific literature (Juels, 2005).

Those protocols were designed to secure communication among legitimate readers and tags against attackers. However, many protocols have been compromised without breaking the cryptographic primitives (e.g., symmetric key, public key, or hash algorithm), instead by simply manipulating the passing messages (Juels, 2005; Li and Deng, 2007). This suggests that there is a flaw in the design of those protocols that is not immediately evident via an cryptographic analysis. We suggest a five step procedure for applying the formal verification methods in the domain of RFID system to the problem of authentication protocol verification. Applying these methods to a class of protocols will allow us to be sure which protocol is logically sound and in fact meets the required objectives. Secondly, stripping those protocols to their necessity and applying the formal methods allow us to directly compare protocols with respect to their similarity and difference. In addition, we provide a guideline to allow protocol designers to distinguish between situations where encryption is required and where a hash function is sufficient.

Burrows et al. (1989) invented BAN logic, which is a modal logic of knowledge and beliefs and uses special constructs to express some of the central concepts in the protocols. It is designed to reason authentication protocols among participants in a distributed computing system. BAN logic started a much needed new research field, the formal verification of authentication protocols. AT logic (Abadi and Tuttle, 1991) constructs a semantics for the BAN logic to capture and clarify its meaning. Gong et al. (1990) further refined and expanded BAN logic to cover a wider range of protocols, which we refer to as GNY logic. The SVO logic (Syverson and Van Oorschot, 2004), which includes temporal formalisms, is also based on beliefs.

GNY logic is to analyse an authentication protocol step by step, make necessary explicit assumptions, use the messages received, apply the logic rules, and draw conclusions about the final goals of the protocol. GNY logic substantially extends BAN logic, introduces the new notions of possession, recognisability, not-originated-here, and the strong separation between the content and the meaning of messages. It can detect a protocol’s subtle errors, flaws, e.g., in the analysis of the famous Needham-Schroeder protocol (Gong et al., 1990). It does not require universal assumptions as BAN logic. It is more complete and robust and can be used to analyse a broad range of protocols. RFID authentication protocols, though limited to lightweight, have been developed based on a wide variety of cryptographic primitives. A complete formal language tool, like GNY logic, is essential to ensure a systematic verification methodology to produce proofs or disproofs.
1.1 Related work

The need for formal verification of cryptographic protocols has come a long way since the 1980s. The cryptographic protocols that have been formally analysed range from voting protocols, smartcard protocols (Bella, 2007) (like Shoup-Rubin) to larger and deployed protocols (like Kerberos V (Butler et al., 2006), Kerberos IV (Bella, 2007)). Formal verification methods include (Bella, 2007) belief logic (Burrows et al., 1989; Gong et al., 1990; Syverson and Van Oorschot, 2004), model checking and verification in process algebras (Lowe, 1996; Schneider, 1998), theorem proving with induction (Bella, 2007), state exploration methods (Meadows, 1994; Mitchell et al., 1997), etc. Just like each of them has its pros and cons. GNY and BAN logic belong to belief logic, which is hard to reason the confidentiality (Bella, 2007). However, in our study on RFID systems, we focus on authentication. Moreover, GNY logic proofs are not sophisticated and can be carried out by hand.

BAN logic has been used to verify the cryptographic protocols like Otway-Rees, Wide-mouthed-frog, Yahalom, Needham-Schroeder, CCITT X.509, Kerberos (Burrows et al., 1989). Recently two papers (Yang et al., 2005; Gódoi and Antal, 2008) use GNY logic to prove the correctness of their proposed RFID authentication protocols. In this paper, we propose and demonstrate a five-step procedure for applying the GNY logic authentication protocol proof to any proposed protocol. We demonstrate our method by verifying the correctness/incorrectness of five recently proposed RFID authentication protocols. We show how our method allows for the determination of when hashing is sufficient or when we must resort to the stronger cryptographic methods.

1.2 Outline of the paper

In this paper we consider a class of recently proposed lightweight RFID authentication protocols, namely CRAP (Rhee et al., 2005), LCAP (Lee et al., 2005), OHLCAP (Choi et al., 2005), O-TRAP (Chatmon et al., 2006), and YA-TRAP (Tsudik, 2006). CRAP stands for Challenge Response Authentication Protocol, LCAP for Low Cost Authentication Protocol, OHLCAP for Our Hashbased Low Cost Authentication Protocol, O-TRAP for Optimistic Trivial RFID Authentication Protocol, and finally YA-TRAP stands for Yet Another Trivial RFID Authentication Protocol. In these protocols, the security models assume that the communication link between backend server and reader is secure. Taking advantage of this assumption, we can further simplify the RFID system and combine backend server and reader into one entity which we call the Reader. We will evaluate these protocols using a modal logic to see if they meet their desired goals.

The rest of this paper is organised as follows. In the next section we briefly introduce basic GNY concepts, syntax, logic rules, and the general proof procedure. In Section 3, we introduce the LCAP protocol and then use the GNY proof procedure (see Section 2.3) to show that the LCAP protocol achieves its protocol goals. In Sections 4 and 5, we examine the protocols CRAP, OHLCAP, O-TRAP, YA-TRAP, and YA-TRAP+ to show whether they attain the basic protocol goals of a one or two-sided protocol as appropriate. Finally, in Section 6, we discuss our conclusions.

2 GNY logic and proof procedure

In this section we briefly introduce basic concepts of GNY logic (Gong et al., 1990) including syntax, reasoning rules, and the general proof procedure for any given authentication protocol.

2.1 Basic concepts and syntax

GNY logic is made up of formulae, communication parties, logic operators, statements, and logical rules. A formula is a bit string which has a specific value in a session of a given protocol. A formula is the smallest reasoning unit in GNY logic. It is generally used to denote an item of information such as a message, a shared secret, or an encryption key. It is represented by a capital letter. By convention X, Y is used for general formulae and S, K for shared secrets and encryption keys. Compositions of formulae, such as the following, are also treated as formulae. \( \langle X, Y \rangle \): concatenation (or conjunction) of two formulae. \( H(X) \): a one-way hash of formula \( X \). \( \{X\}_K \) and \( \{X\}^{-1}_K \): conventional encryption and decryption of formula \( X \) with the key \( K \).

A communication party is a participant in a protocol. Though also denoted by a capital letter, it can be easily discerned from formulae from the context in which it is used. \( P, Q \) will be used for general parties and \( R, T \) will be used for Reader and Tag in the RFID protocols.

A logic operator is a symbol that performs or describes a logic operation or property, such as \( \prec \), \( \equiv \), \( \rhd \) (see the statements below). Operators are used in expressions (statements) to associate parties and formulae. A statement, denoted by an expression, is used to describe certain properties of formulae and parties. The following are some basic statements. Here \( P, Q \) are parties, \( X \) is a formula, and \( S, K \) are formulae for shared secret and encryption key.

- \( P \prec X \) : \( P \) receives \( X \), including possibly encrypted version of it, if \( P \) is able to decrypt (\( \prec \) is the receiving operator).
- \( P \prec \ast X \) : \( P \) receives \( X \) which is not originated from \( P \) itself. \( \ast \) is ‘not-originated-here’ operator.
- \( P \rhd X \) : \( P \) possesses, or is able to possess \( X \) (here \( \rhd \) is the possessing operator).
- \( P \rhd X \) : \( P \) once conveyed or said \( X \), explicitly or implicitly (\( \rhd \) is has-conveyed operator).
\begin{itemize}
  \item \[P \models \varphi(X)\]: \(P\) believes (\(\equiv\) is the believing operator) that \(X\) is fresh (\(\varphi()\) is the is-fresh operator). I.e., \(X\) has not been used at any previous sessions of the protocol.
  \item \[P \models \phi(X)\]: \(P\) believes that \(X\) is recognisable (\(\phi()\) the is-recognisable operator). I.e., the (partial) contents of \(X\) are recognisable for \(P\) even before it actually receiving it.
  \item \[P \models P \overset{\delta}{\leftrightarrow} Q\]: \(P\) believes that \(S\) is a suitable shared secret for \(P\) and \(Q\), and only known to them (\(\overset{\delta}{\leftrightarrow}\) is the sharing operator).
  \item \[P \models Q \Rightarrow C\]: \(P\) believes that \(Q\) has jurisdiction (authority) over a statement \(C\) (\(\Rightarrow\) is the has-jurisdiction operator).
  \item \[P \models Q \Rightarrow Q \Rightarrow *\]: \(P\) believes that \(Q\) has jurisdiction (authority) over all \(Q\)’s beliefs (here * is the wildcard, means everything).
  \item \[X \leadsto C\]: Statement \(C\) is an extension of formula \(X\). \(C\) is an implicit true assumption statement easily inferred from the formula \(X\) (\(\leadsto\) is the has-extension operator).
  \item \((C_1, C_2, \ldots, C_i)\): The concatenation of \(i\) statements \(C_1, C_2, \ldots, C_i\) is also a statement.
\end{itemize}

\subsection{Logic rules}

GNY logic postulates inference rules that are split into six categories: being-told rules, possession rules, freshness rules, interpretation rules, jurisdiction rules, and recognisability rules. Each rule is labelled with rule’s category initial followed by a serial number, e.g., P3 represents the third possession rule (we will use the same numbering scheme for other statements as well later in the paper). Rules are expressed in a big fraction format with the top and bottom statements separated by a horizontal line. It reads, if the top statement holds, then the bottom statement follows. Here we give the rules used in this paper, the complete list of rules can be found in Gong et al. (1990).

\begin{itemize}
  \item \[P \triangleleft \{X\}_K, P \triangleright K, P \models P \overset{\delta}{\leftrightarrow} Q, P \models \phi(X), P \models \varphi(X, K)\]
  \item \[P \triangleleft \{X\}_K, P \triangleright K\]
  \item \[P \models Q \leadsto X, P \models \varphi(X)\]
  \item \[P \models Q \leadsto Q \leadsto *\]
  \item \[P \models Q \leadsto C, P \models Q \leadsto *\]
  \item \[P \models Q \leadsto Q \leadsto *\]
  \item \[P \models \varphi(X)\]
  \item \[P \models \varphi(X, Y)\]
\end{itemize}

\section{LCAP: low cost RFID authentication protocol}

Lee et al. (2005) proposed a cost effective RFID authentication protocol, called LCAP, which needs only two one-way hash function operations. In each authentication session the tag identifier, used as a shared secret with the reader, is dynamically updated and the
tag response is randomised. The protocol achieves mutual authentication and prevents spoofing, impersonation, and tracing attacks.

3.1 Protocol description

Figure 1 is the conventional way to describe a protocol with Reader and Tag as two communication parties and the messages passing between them. In Figure 1, $H$ is a one-way hash function. $H_L$, $H_R$ are the left and the right halves of the hash value. $N_i$ is the nonce generated by Reader at $i$th session. Both Reader and Tag update $ID$ by $ID_{i+1} = ID_i \oplus N_i$.

![Figure 1: LCAP protocol in diagram](Image)

Let $T$ represent the Tag, $R$ represent the Reader, and $S_i$ represent the secret identifier in session $i$, $ID_i$. First we list the protocol messages of the generic type as follows (here M1, M2 and M3 stand for the first, second and third message):

M1 $R \rightarrow T : N_i$
M2 $T \rightarrow R : H(S_i), H_L(S_i, N_i)$
M3 $R \rightarrow T : H_R(S_i, N_i)$

For the $i$th authentication session the reader $R$ picks a random nonce $N_i$ and sends it to tag $T$ as M1. The tag computes $H(S_i)$ and $H(S_i, N_i)$ by using $N_i$ and its own current secret $ID_i$. $T$ sends $H(S_i)$ and $H_L(S_i, N_i)$ to $R$ as M2, where $H_L(S_i, N_i)$ is a left half of $H(S_i, N_i)$. Then $R$ checks that $H(S_i)$ matches a record in the database. If successful, the $R$ authenticates the $T$ by computing $H(S_i, N_i)$ and sending the right half of it, $H_R(S_i, N_i)$, back to $T$ as M3. On $T$ side, if the received $H_R(S_i, N_i)$ matches the locally computed counterpart, $T$ authenticates $R$.

3.2 Protocol goals and assumptions

The LCAP protocol messages can be expressed using GNY logic statements as follows:

M1 $T \leftarrow \ast N_i$
M2 $R \leftarrow \ast H(S_i), \ast H_L(N_i, < S_i \gg \sim (T \equiv T \uparrow \downarrow R))$
M3 $T \leftarrow \ast H_R(N_i, < S_i \gg \sim (R \equiv R \uparrow \downarrow T))$

3.2.1 The goals

The LCAP protocol goal is to offer mutual authentication to $R$ and $T$. The desired final position at the end of an CAP authentication session can be formalised in the GNY logic as the following beliefs:

- **G1** $R \equiv T \equiv S_i$
- **G2** $R \equiv T \equiv T \uparrow \downarrow R$
- **G3** $T \equiv R \equiv S_i$
- **G4** $T \equiv R \equiv R \uparrow \downarrow T$

G1 states $R$ is confident that $T$ is in possession of the new secret $S_i$. G2 states $R$ is confident that $T$ believes only the two of them know and share the secret $S_i$. Likewise G3 states $T$ is confident that $R$ is in possession of the new secret $S_i$, and G4 states that $T$ is confident that $R$ believes only the two of them know and share $S_i$.

3.2.2 The assumptions

The following statements represented by $A$ followed by a number are the assumptions which are assumed true in the LCAP protocol. These assumptions will help us to prove the above goals using GNY inference rules.

- **A1** $R \equiv S_i$
- **A2** $R \equiv N_i$
- **A3** $R \equiv \ast (N_i)$
- **A4** $R \equiv R \uparrow \downarrow T$
- **A5** $R \equiv T \equiv T \uparrow \sim$
- **A6** $R \equiv \ast (S_i)$
- **A7** $T \equiv S_i$
- **A8** $T \equiv T \uparrow \downarrow R$
- **A9** $T \equiv R \equiv R \uparrow \downarrow T$
- **A10** $T \equiv \ast (S_i)$

Here A1 and A2 mean that $R$ possesses secret $S_i$ and random nonce $N_i$ at the beginning of the $i$th authentication session. A4 says that $R$ believes that $S_i$ is a suitable secret only known to $T$ and itself. A5 says that $R$ believes that $T$ has jurisdiction over all $T$'s beliefs. The party $T$ is considered by $R$ to be honest, competent and trustworthy. Similar explanation applies to the assumption A9. A6 says that $R$ believes that the secret $S_i$ is fresh. $S_i$ is fresh because of the freshness of $N_{i-1}$ (the updating is $S_i = S_{i-1} \oplus N_{i-1}$).

3.3 Protocol proof

By using the assumptions A1~A9, formalised messages M1~M3 and the GNY logic statements, we are able to deduce the protocol goals. We label the intermediate results with D (for ‘Deduction’) followed by a serial number.

On receiving M1 $T \leftarrow \ast N_i$, we have

- **D1**. $T \leftarrow N_i$ /* By M1, T1 */

On receiving M2 $R \leftarrow \ast H(S_i), \ast H_L(N_i, < S_i \gg \sim (T \equiv T \uparrow \downarrow R))$, we have

- **D2**. $T \equiv N_i$ /* By D1, P1 */
3. **R** \( \triangleleft \star H_L(N_i, S_i) \sim (T \equiv T \overset{\leftarrow}{R}) \) /* By M2, T2 */

4. **R** \( \equiv \xi(N_i, S_i) \) /* By A3, F1 */

5. **R \triangleright (N_i, S_i) */ By A1, A2, P2 */

By using D3, D5, A4, D4, and I3 we can have the temporary result

\[
R \triangleleft \star H_L(N_i, S_i), R \triangleright (N_i, S_i) R \equiv \overline{R} \xi(R, S_i) T, R \equiv \xi(N_i, S_i)
\]

6. **R \equiv T \vdash (N_i, S_i) */ By above result */

7. **R \equiv T \triangleright (N_i, S_i) */ By M3, D10, A8, D11, I3 */

8. **R \equiv T \triangleright S_i */ By D8, D4, I6 */

9. **R \equiv T \triangleright T \overline{\overset{\leftarrow}{R}} */ By A5, D6, A6, J2 */

We see that D8, D9 are the goals G1, G2.

On receiving M3 \( T \triangleleft \star H_R(N_i, S_i) \sim (R \equiv R \overline{\overset{\leftarrow}{H}} T) \), we have

10. **T \triangleright (N_i, S_i) */ By D2, A7, P2 */

11. **T \equiv \xi(N_i, S_i) */ By A10, F1 */

12. **T \equiv R \triangleright (N_i, S_i) \sim (R \equiv R \overline{\overset{\leftarrow}{H}} T) */ By M3, D10, A8, D11, I3 */

13. **T \equiv R \triangleright S_i */ By D12, D11, I6 */

14. **T \equiv R \triangleright S_i */ By D13, P3 */

15. **T \equiv R \triangleright (S_i) \sim (R \equiv R \overline{\overset{\leftarrow}{H}} T) */ By D12, I7 */

16. **T \equiv R \equiv R \overline{\overset{\leftarrow}{H}} T */ By A9, D15, A10, J2 */

17. **T \equiv R \equiv R \overline{\overset{\leftarrow}{H}} T */ By D16, J3 */

We see that D14, D17 are the goals G3, G4.

### 3.3.1 Remarks

Using GNY logic, we are able to prove that the RFID authentication protocol LCAP achieves the expected goals that both reader and tag are confident in the belief that they are talking to the intended parties.

### 4 OHLCAP and CRAP protocols

Choi et al. (2005) proposed another LCAP, called OHLCAP, for Our Hash-based Low Cost Authentication Protocol. OHLCAP is suitable for ubiquitous computing environment where RFID components could exist anywhere. It is efficient because it involves only one hash operation, four exclusive-or (XOR), and one addition operation. Rhee et al. (2005) proposed CRAP, Challenge-Response Authentication Protocol, for RFID applications in a distributed database environment. CRAP uses a one-way hash function and a random number. It is secure against the replay, spoofing, and traffic analysis attacks with strong anonymity.

### 4.1 Logic reasoning of CRAP

Figure 2 shows how the CRAP protocol works, in which \( H(\cdot) \): one-way hash function, \( N_R \): nonce generated by Reader, \( N_T \): nonce generated by Tag, ID: Tag secret identification shared by Reader and Tag. The Reader challenges the Tag by sending a random nonce. Then the Tag responds to it using the received random nonce and another self generated random nonce. The CRAP protocol does not update the shared secret identifier (ID). This aspect is different from the hash-based ID variation protocol (Henrici and Muller, 2004), a predecessor of CRAP. It is also different from the LCAP protocol (Lee et al., 2005) that we analysed in the previous section.

#### 4.1.1 Protocol description

In session \( i \),

1. **Reader** \( R \) broadcasts a random nonce \( N_{R_i} \) as a challenge to \( T \).

2. **Tag** \( T \) generates a random nonce \( N_{T_i} \), hashes the concatenation of \( ID, N_{R_i} \) and \( N_{T_i} \), and then sends the hash together with \( N_{T_i} \) to \( R \) as the response.

3. For each \( ID \) stored in the Reader’s database, the ID is concatenated with \( N_{R_i} \) and \( N_{T_i} \), and the result is hashed. \( R \) compares its computed hashes with the received \( H(ID, N_{R_i}, N_{T_i}) \) from \( T \). If a match is found, then \( R \) authenticates \( T \) and sends \( H(ID, N_{T_i}) \) back to \( T \).

4. **Tag** \( T \) receives \( H(ID, N_{T_i}) \) from \( R \) and compares it with the computed \( H(ID, N_{T_i}) \). If they match, then \( T \) authenticates \( R \). The authentication session \( i \) is complete.

To avoid having to include a separate random nonce generator on the Tag \( T \), an improved version of CRAP uses the \( H(\cdot) \) as a random nonce generator in the following way. \( N_{T_i} = H(key), N_{T_i}^2 = H(key, N_{T_i}^1, N_{R_i}^{i-1}) \), here \( key \) is another piece of secret information embedded on \( T \) when it was made.
4.1.2 Protocol assumptions and goals

Let $S$ represent ID. In the CRAP protocol the transfer of messages in generic format can be expressed as:

- **M1**  $R \rightarrow T$: $N_R^i$
- **M2**  $T \rightarrow R$: $H(S, N_R^i, N_T^i, N_T^i)$
- **M3**  $R \rightarrow T$: $H(S, N_T^i)$.

And the messages formalised in the format of GNY logic statements are:

- **M1**  $T \leftarrow \ast N_R^i$
- **M2**  $R \leftarrow \ast H(N_R^i, \ast N_T^i, \langle S \rangle \leadsto (T \equiv T \overleftarrow{S} \rightarrow R))$, $N_T^i$
- **M3**  $T \leftarrow \ast H(N_T^i, < S > \leadsto (R \equiv R \overleftarrow{S} \rightarrow T))$.

Assumptions of the CRAP protocol are listed here with labels that match similar assumptions of the earlier LCAP proof.

- **A1**  $R \ni S$
- **A2**  $R \ni N_R^i$
- **A3**  $R \equiv \overline{\ast} (N_R^i)$
- **A4**  $R \equiv R \overleftarrow{S} \rightarrow T$
- **A5**  $T \ni T \equiv \ast$
- **A6**  $R \equiv \overline{\ast} (S)$ Not hold!
- **A7**  $T \ni S$
- **A8**  $T \equiv T \overleftarrow{S} \rightarrow R$
- **A9**  $T \ni N_T^i$
- **A10**  $T \equiv \overline{\ast} (N_T^i)$
- **A11**  $T \equiv R \rightarrow R \equiv \ast$
- **A12**  $T \equiv \overline{\ast} (S)$ Not hold!

Notice that, except for A6 and A12, the other assumptions are almost identical to the assumptions made in Section 3.2 for the proof of LCAP protocol. The freshness of the shared secret $S$ cannot be guaranteed, because the definition of freshness is that $S$ cannot be used in more than one session for the same purpose. Because CRAP is proposed as a mutual authentication protocol, the same goals as the LCAP protocol should be achieved at the end of each authentication session, they are:

- **G1**  $R \equiv T \ni S$
- **G2**  $R \equiv T \equiv T \overleftarrow{S} \rightarrow R$
- **G3**  $T \equiv R \ni S$
- **G4**  $T \equiv R \equiv R \overleftarrow{S} \rightarrow T$.

4.1.3 Protocol proof

We follow the general proof procedure outlined in Section 2.3 to attempt to deduce the protocol goals using the above assumptions, formalised message passing statements, and GNY inference. Again, we label the intermediate statements during the deduction process with D followed by a serial number.

On receiving **M1**  $T \leftarrow \ast N_R^i$, we have

- **D1**  $T \leftarrow N_R^i$ /* By M1, T1 */
- **D2**  $T \ni N_R^i$ /* By D1, P1 */

On receiving **M2**  $R \leftarrow \ast H(N_R^i, \ast N_T^i, \langle S \rangle \leadsto (T \equiv T \overleftarrow{S} \rightarrow R))$, $N_T^i$, we have

- **D3**  $R \leftarrow \ast H(N_R^i, \ast N_T^i, \langle S \rangle \leadsto (T \equiv T \overleftarrow{S} \rightarrow R))$/* By M2, T2 */
- **D4**  $R \equiv N_T^i$ /* By M2, T2 */
- **D5**  $R \ni (N_R^i, N_T^i, S)$ /* By A1, A2, D4, P2 */
- **D6**  $R \equiv \overline{\ast} (N_R^i, N_T^i, S)$ /* By A3, F1 */

By using D3, D5, A4, D6, and I3 we can have the temporary result

$$R \ni \ast H(N_R^i, \ast N_T^i, \langle S \rangle \leadsto (T \equiv T \overleftarrow{S} \rightarrow R)) \ni \ast H(N_R^i, N_T^i, \langle S \rangle \leadsto (T \equiv T \overleftarrow{S} \rightarrow R))$$

/* By above result */

$$R \equiv T \ni (N_R^i, N_T^i, \langle S \rangle \leadsto (T \equiv T \overleftarrow{S} \rightarrow R))$$

/* By above result */

$$R \equiv T \ni (N_R^i, N_T^i, \langle S \rangle \leadsto (T \equiv T \overleftarrow{S} \rightarrow R))$$

/* By D7, D6, I6 */

$$R \equiv T \ni S \ast \equiv \ast$$/* By D8, P3 */

We know that A5 and D7 hold, but A6, the freshness assumption $R \equiv \overline{\ast} (S)$, does not hold. Therefore we cannot use rule J2 to deduce $R \equiv T \equiv T \overleftarrow{S} \rightarrow R$. Thus, we cannot prove that the goal, G2, is true by the end of the message M2. However, the goal G1 can be deduced from D9.

On receiving **M3**  $T \leftarrow \ast H(N_T^i, \langle S \rangle \leadsto (R \equiv R \overleftarrow{S} \rightarrow T))$, we have

- **D10**  $T \ni (N_T^i, S)$ /* By A7, A9, P2 */
- **D11**  $T \equiv \overline{\ast} (N_T^i, S)$ /* By A10, F1 */
- **D12**  $T \equiv \overline{\ast} (N_T^i, \langle S \rangle \leadsto (R \equiv R \overleftarrow{S} \rightarrow T))$/* By M3, D10, A8, D11, I3 */
- **D13**  $T \ni R \ni (N_T^i, S)$ /* By D12, D11, I6 */
- **D14**  $T \ni R \ni S$ /* By D13, P3 */

Again, although A11 and D12 hold, the freshness assumption $T \equiv \overline{\ast} (S)$ does not hold (A12), so we cannot use J2 to deduce $T \equiv R \equiv R \overleftarrow{S} \rightarrow T$. Therefore, the goal G4 cannot be proved by the end of M3, though the goal G3 is reached from D14.

4.1.4 Remarks

A party’s belief in the freshness of a shared secret represents his belief that the secret has not been used at any time before the current session of the protocol (Gong et al., 1990). In the CRAP protocol, because the shared secret $S$ (i.e., ID) does not change for different sessions (it is static), it has been used in every session, therefore $S$ is not fresh. So the jurisdiction rule (J2) cannot be used to infer two of the protocol goals. The communication parties Reader and Tag are unable to convince one another that $S$ is a suitable secret only known to them.
4.2 Logic of OHLCAP

After original OHLCAP was proposed by Choi et al. (2005) and Ha et al. (2007) enhanced OHLCAP by removing the counter used in the original protocol. Because the counter causes a distinguishable sign in one of the messages, it makes the tag location traceable and the impersonation attack possible. Here we will carry out the logic analysis for the enhanced OHLCAP. In the following analysis we ignore the Group Index (GI) secret and one of the Tag response messages, A2 since it is a function of GI, ID and the challenge nonce.2 Further, we drop the modular addition operation from our consideration. The final stripped version of the OHLCAP is depicted in Figure 3, in which the scheme consists of the passing messages among the Reader R and the Tag T. The assumptions are the same as the CRAP protocol. Using the same deduction process as in CRAP protocol, we are able to prove the same results as in Section 4.1, i.e., at the end of the session i the goals G1 and G3 can be Reached. However, it is not possible to reach G2 or G4. In both cases it is because the freshness of the shared secret S cannot be guaranteed. We skip the proof procedure due to the significant overlap with the proof for CRAP protocol.

4.2.1 Protocol description

Initially both Reader R and Tag T are installed with the shared secret (ID, K). K is a secret bit-string stored in all tags. The authentication session i consists of the following steps.

1. R sends a random nonce N∗_R_i to tag T.
2. T generates another random nonce N∗_T_i and computes A′ = K ⊕ N∗_T_i and H(ID, N∗_R_i, N∗_T_i). T then sends A′, H_R(ID, N∗_R_i, N∗_T_i) to R.
3. R computes N′_T_i = A′ ⊕ K and looks for the ID in its database by checking the H(ID, N′_R_i, N′_T_i). If the computed H_R(ID, N′_R_i, N′_T_i) equals the received one, R has successfully authenticated T. R then sends the H_L(ID, N′_R_i, N′_T_i) to T.
4. T is able to authenticate R if the received H_L(ID, N′_R_i, N′_T_i) equals the left half computed in step (2).

4.2.2 Message formalisation

Let S represent ID. The passing messages in OHLCAP are:

M1 R → T: N∗_R_i
M2 T → R: A′, H_R(S, N∗_R_i, N∗_T_i)
M3 R → T: H_L(S, N′_R_i, N′_T_i)

And by ignoring A′ (not significant) the messages formalised in the format of GNY logic statements are:

M1 T ⊥ N∗_R_i
M2 R ⊥ H_R(N′_R_i, N′_T_i, ⟨S⟩) ≈ (T ⊥ T S ≈ R)
M3 T ⊥ H_L(N∗_R_i, N∗_T_i, ⟨S⟩) ≈ (R ≈ R T S ≈ T)

4.2.3 Protocol proof

From the protocol description and the formalised passing messages, OHLCAP is very similar to CRAP. For the assumptions and the protocol goals we refer to Section 4.1, because both OHLCAP and CRAP share the same assumptions and goals.

Although OHLCAP exchanges slightly different messages among the Reader R and the Tag T, the assumptions are the same as the CRAP protocol. Using the same deduction process as in CRAP protocol, we are able to prove the same results as in Section 4.1, i.e., at the end of the session i the goals G1 and G3 can be Reached. However, it is not possible to reach G2 or G4. In both cases it is because the freshness of the shared secret S cannot be guaranteed. We skip the proof procedure due to the significant overlap with the proof for CRAP protocol.

4.2.4 Remarks

Our analysis shows that the OHLCAP and CRAP protocols share the same deficiency. They do not enable both communication parties to convince one another that the shared secret is suitable and only known to them. The cause of that is the use of a static ID as the shared secret, and thus the failure to satisfy the freshness assumption of the secret. Meanwhile from the analysis in Section 3, we can see that the LCAP protocol achieves all its goals. In this sense, the enhanced OHLCAP is inferior to the original LCAP.

5 O-TRAP, YA-TRAP and YA-TRAP+ protocols

TRAP stands for Trivial RFID Authentication Protocol. It has several variations. One common trick of the various TRAPs is to place as light as possible a computational burdens on the tag, while shifting heavy computational burdens on to the reader. Although there are two-sided, i.e., mutually authenticated, TRAPs, like A-TRAP (‘A’ for Absolutely) (Le et al., 2007), here we will consider three one-side TRAPs, in which only the reader authenticates the tag. Chatmon et al. (2006) proposed the O-TRAP (‘O’ for Optimistic). They called it a 1-pass protocol. However we consider the broadcasting of the challenge by reader R to all tags as an extra message pass, so we treat it as a 2-pass protocol. With this protocol RFID tags are authenticated with minimal security overhead when the parties are honest. Burmester et al. (2006) proved the security
of the O-TRAP under the universal composable-type adversary model. Tsudik (2006) proposed YA-TRAP. Yet Another TRAP. YA-TRAP involves minimal interaction between a tag and a reader. It uses monotonically increasing timestamps to provide tag anonymity. However, YA-TRAP is susceptible to the trivial Denial-of-Service (DoS) attack. Shortly after (Chatmon et al., 2006) proposed YA-TRAP+, an extended version of YA-TRAP. YA-TRAP+ is secure against DoS attacks and its other security features are proved in Burmester et al. (2006).

5.1 Logic of O-TRAP

5.1.1 Protocol description

Initially tags are embedded with individual unique secret keys \((K_j, N_{K_j})\) which they share only with the authorised reader. \(R\). \(K_j\) serves as the secret ID and \(N_{K_j}\) as the initial nonce for the tag \(j\). On the reader \(R\) side, the protocol uses a key lookup table in which each tag \(T_j\) is associated with the shared secret key \(K_j\) and a session nonce \(N_{K_j}\), see Table 1.

<table>
<thead>
<tr>
<th>Tags</th>
<th>(T_1)</th>
<th>(T_2)</th>
<th>\ldots</th>
<th>(T_n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keys</td>
<td>(K_1)</td>
<td>(K_2)</td>
<td>\ldots</td>
<td>(K_n)</td>
</tr>
<tr>
<td>Strings</td>
<td>(N_{K_1})</td>
<td>(N_{K_2})</td>
<td>\ldots</td>
<td>(N_{K_n})</td>
</tr>
</tbody>
</table>

For a particular tag \(T\) with the initial shared secret \((K, N^0_K)\), in session \(i\) the reader \(R\) sends (broadcasts) a random nonce \(N^i_R\) to \(T\) as a challenge. Upon receiving \(N^i_R\), \(T\) concatenates it with its locally generated nonce \(N^i_K\), computes the \(H_K(N^i_R, N^i_K)\), and then sends the result back to \(R\) as its response. The two messages are:

M1 \(R \rightarrow T: N^i_R\)

M2 \(T \rightarrow R: H_K(N^i_R, N^i_K)\)

Here \(H_K(\ )\) is a keyed hash. \(T\) updates the \(N^i_K\) by \(N^{i+1}_K = H_K(N^i_R, N^i_K)\) for the next session. \(R\) authenticates \(T\) if:

- (Case 1) there exists a pair \((K, N^i_K)\) in the key lookup table such that \(H_K(N^i_R, N^i_K)\) matches the received one (optimistic)
- (Case 2) try every key \(K\) to compute \(H_K(N^i_R, N^i_K)\) and a match is found.

Case 2 will occur only if \(T\) has recently been interrogated by an unauthorised reader which has set the \(T\) side \(N^i_K\) out-of-sync with the \(R\) side \(N^i_K\). The re-sync procedure is done on the \(R\) by advancing each \(N^i_K\) in the key lookup table to compute \(H_K(N^i_R, N^i_K)\) until a match with that received from \(R\) is reached. After \(T\) is authenticated, \(R\) updates \(N^i_K = H_K(N^i_K)\) for the next session.

5.1.2 Protocol assumptions and goals

In our analysis we will treat keyed hash \(H_K(X)\) as an encryption \(\{X\}_K\) with a secret key \(K\). For O-TRAP the messages formalised as GNY logic statements are:

M1 \(T \leftarrow *N^i_R\)

M2 \(R \leftarrow *\{N^i_R, *N^i_K\}_K \sim \to (T \equiv T \leftrightarrow R)\)

The assumptions for O-TRAP are listed as follows. In this one-side authentication protocol only the \(R\) authenticates the \(T\), thus the \(T\) side assumptions do not matter and we ignore them.

A1 \(R \ni K\)

A2 \(R \equiv \not\exists(N^i_R)\)

A3 \(R \equiv R^i_K\)

A4 \(R \equiv \phi(N^i_K)\)

A5 \(R \ni N^i_R\)

A6 \(R \equiv T \vdash T \equiv *\)

The goals of the O-TRAP protocol should be achieved at the end of an authentication session, they are:

G1 \(R \equiv T \ni K\)

G2 \(R \equiv T \equiv T^i_K \leftrightarrow R\)

That is to say at the end of a session \(R\) is confident that \(T\) is in possession of the shared secret \(K\), and \(R\) is confident that \(T\) believes only two of them know and share the secret \(K\).

5.1.3 Protocol proof

Again, we label the intermediate statements with \(D\) followed by a serial number. Using the assumptions, formalised messages, and GNY inference rules, we try to deduce the protocol goals. The proof procedure progresses as below.

On receiving \(M2\) \(R \leftarrow *\{N^i_R, *N^i_K\}_K \sim \to (T \equiv T \leftrightarrow R)\), we have

D1. \(R \equiv \phi(N^i_R, N^i_K)\) /* By A4, R1 */

D2. \(R \equiv \not\exists(N^i_R, N^i_K)\) /* By A2, F1 */

By using M2, A1, A3, D1, D2 and II we can have

\[
R \leftarrow *\{N^i_R, *N^i_K\}_K, R \ni K, R \equiv R^i_K, T, R \equiv \phi(N^i_R, N^i_K), R \equiv \not\exists(N^i_R, N^i_K), T \equiv T \leftrightarrow (N^i_R, N^i_K), R \equiv T \ni T \equiv D3. \equiv * By above result */

D4. \(R \equiv T \equiv (N^i_R, N^i_K, K) \leftrightarrow (T \equiv T \leftrightarrow R)\) /* By above result */

D5. \(R \equiv \not\exists(*N^i_R, *N^i_K)\) /* By D2, A1, F2 */

D6. \(R \equiv T \equiv T \leftrightarrow R\) /* By A6, D4, D5, J2 */

We see that D3, D6 are the goals G1, G2.
5.1.4 Remarks

Although a static key (or ID) is used as the shared secret, the protocol uses a keyed hash to manipulate the key and random nonce to create messages, instead of a simple hash. In the proof we treat the keyed hash as an encryption with that secret key. From the above analysis we see that the O-TRAP has achieved the intended goals.

5.2 Logic of YA-TRAP+

5.2.1 Protocol description

Suppose that initially tags are embedded with unique secret keys $K_j$ which they share only with the authorised reader $R$. Assume also that tags are able to generate a random nonce. On the reader side the protocol uses a hash lookup table Table 2 in which each tag, $R_i$ is associated with the tag’s hash value at session $i$ and timestamps $t_R^i$. Here $H^i_1 = H_{K_j}(0, t_R^i, N_R^i)$, and $H_K(\cdot)$ is the keyed hash.

Table 2 Hash lookup table

<table>
<thead>
<tr>
<th>Tags</th>
<th>$T_1$</th>
<th>$T_2$</th>
<th>...</th>
<th>$T_n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>keys</td>
<td>$K_1$</td>
<td>$K_2$</td>
<td>...</td>
<td>$K_n$</td>
</tr>
<tr>
<td>$t_R^1$</td>
<td>$H^1_1$</td>
<td>$H^1_2$</td>
<td>...</td>
<td>$H^1_n$</td>
</tr>
<tr>
<td>$t_R^2$</td>
<td>$H^2_1$</td>
<td>$H^2_2$</td>
<td>...</td>
<td>$H^2_n$</td>
</tr>
</tbody>
</table>
| ... | ... | ... | ... | ...
| $t_R^n$ | $H^n_1$ | $H^n_2$ | ... | $H^n_n$ |

The passing messages between $R$ and $T$ of YA-TRAP+ are as follows:

M1 $R \rightarrow T$: $(t_R^i, N_R^i)$

M2 $T \rightarrow R$: $N_T^i, H_K(0, t_R^i, N_R^i)$ if $t_R^i > t_T$

M2’ $T \rightarrow R$: $N_T^i, H_K(1, N_T^i, N_R^i)$ if $t_R^i \leq t_T$

$t_R^i$ and $t_T^i$ are timestamps from the reader $R$ and the tag $T$. $R$ authenticates $T$ if there exists a $K$ in the key space such that $H_K(0, t_R^i, N_R^i)$ or $H_K(1, N_T^i, N_R^i)$ can be found in Table 2. After successful authentication, $T$ updates its timestamps by $t_T^{i+1} = t_R^i$.

5.2.2 Message formalisation

We treat keyed hash $H_K(X)$ as an encryption $\{X\}_K$ with key $K$. The messages formalised as GNY logic statements are:

M1 $T \triangleleft \ast(t_R^i, N_R^i)$

M2 $R \triangleleft \ast N_T^i, \ast(0, t_R^i, N_R^i)_K \sim (T \equiv T K R)$

M2’ $R \triangleleft \ast N_T^i, \ast(1, N_T^i, N_R^i)_K \sim (T \equiv T K R)$

5.2.3 Protocol proof

YA-TRAP+ is another one-side authentication protocol that shares the same assumptions and protocol goals with the O-TRAP protocol in Section 5.1. We prove the goals are reached once the message M2 is received.

On receiving $M2: R \triangleleft \ast N_T^i, \ast(0, t_R^i, N_R^i)_K \sim (T \equiv T K R)$

D1. $R \equiv \phi(0, t_R^i, N_R^i)$ /* By A4, R1 */

D2. $R \equiv \sharp(0, t_R^i, N_R^i)$ /* By A2, F1 */

By using M2, A1, A3, D1, D2 and II we can have

\[ R \triangleleft \ast(0, t_R^i, N_R^i)_K, R \equiv K, R \equiv R K T, \]

\[ R \equiv \phi(0, t_R^i, N_R^i), R \equiv \sharp(0, t_R^i, N_R^i) \]

\[ R \equiv T \sim(0, t_R^i, N_R^i), R \equiv T \equiv K, \]

\[ R \equiv T \sim \ast(0, t_R^i, N_R^i)_K, \]

D3. $R \equiv T \equiv K$ /* By above result */

D4. $R \equiv T \sim \ast(0, t_R^i, N_R^i)_K \sim (T \equiv T K R)$ /* By above result */

D5. $R \equiv \sharp(\ast(0, t_R^i, N_R^i)_K)$ /* By D2, A1, F2 */

D6. $R \equiv T \equiv T K R$ /* By A6, D4, D5, J2 */

On receiving $M2\': R \triangleleft \ast N_T^i, \ast(1, N_T^i, N_R^i)_K \sim (T \equiv T K R)$, by following the same steps as done for the reception of M2, except replacing $(0, t_R^i, N_R^i)$ with $(1, N_T^i, N_R^i)$, we end up with the same results. We can see that D3, D6 are the goals G1, G2.

5.2.4 Remarks

This one-side authentication protocol also uses a static key (or ID) as the shared secret between communication parties and a keyed hash to manipulate the key and random nonce to generate messages. We treat the keyed hash as an alternative encryption algorithm. We also conclude that YA-TRAP+ has achieved the expected protocol goals.

6 Conclusions

Using the GNY inference rules and the general proof procedure we have analysed a class of RFID Authentication Protocols (APs), namely CRAP, LCAP, OHLCAP, O-TRAP, YA-TRAP, and YA-TRAP+. We show that the most of these protocols meet their goals, though some do not. We summarise the proof results of those protocol in Table 3, in which we use the following notation: D – Dynamic, S – Static, H – Hash, E – Encryption, A – Achieved, NA – Not Achieved, NFS – Not Fresh Secret. Furthermore this approach has enabled us to identify similarities and subtle differences among these protocols.
From the proofs of the LCAP and OHLCAP protocols we can see that the former can achieve all four goals while the latter fails to convince both the reader and tag that the shared secret is suitable and only known to them. In this sense, the OHLCAP is worse than the original LCAP. The cause of the OHLCAP’s failure is the use of a static shared secret. It thus fails to meet the freshness assumption of the secret.

Comparing the proofs of OHLCAP and O-TRAP, we can see that they both use a static shared secret for the authentication. But the O-TRAP protocol achieves its goals. This is because O-TRAP uses the shared secret as the key to encrypt the secret along with random nonce to create messages, while OHLCAP hashes the shared secret along with a random nonce to create messages. We can see that when an authentication protocol uses static shared secret, encryption can be used to achieve the protocol goals. Meanwhile, when a protocol uses dynamic shared secret, hash operations seem to be good enough to achieve the protocol goals.

Acknowledgements

Authors would like to thank Michael Anshel and Jerry Waxman for valuable discussions. We greatly appreciate Michelle Baker for her encouragement and careful editing of the first draft. We also would like to express our thanks to the reviewers for the helpful suggestions and comments. X. Zhang was supported in part by the PSC-CUNY Research Award: PSCOOC-39-210.

References


Notes

1 Interestingly these protocols’ names all end with the suffix, AP.

2 Do not confuse the Tag response message A2 with assumption statement A2.